UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/827,523	04/20/2004	Keerthi Bhushan K N	200400479-2	2763

22879 7590 03/18/2008

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
SMITH, CHENECA	

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
03/18/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
mkraft@hp.com
ipa.mail@hp.com

Office Action Summary	Application No.	Applicant(s)	
	10/827,523	K N ET AL.	
	Examiner	Art Unit	
	Cheneca P. Smith	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-33* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-33* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *20 April 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Remarks

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on November 9, 2007 has been entered.

2. Applicant's arguments with respect to claims 1-33 have been considered but are moot in view of the new ground(s) of rejection – see Sachs et al (US Patent 5,560,028), art made of record for maintaining and assigning a group number to each instruction.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 13-32 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 13 recites an "apparatus" that has been reasonably interpreted as computer program, software, listing per se (see FIG.1 and pages 4-6 of specification). Claim 13 fails to recite the "apparatus" as stored on an appropriate computer readable medium, which defines structural and functional interrelationships between the software and other components of a computer that

permit the software's functionality to be realized - see MPEP 2106.01(I). Therefore, claim 13 is rejected as non-statutory.

Claims 14-24 mirror the deficiencies of claim 13 and are also rejected as non-statutory.

Claim 25 recites a "computer-readable medium." However, it appears that this "computer-readable medium" is intended to include "transmission means," which could be a signal (see page 18 lines 18-20 of specification). A product is a tangible physical article or object, some form of matter, which a signal is not. A signal, a form of energy, does not fall within one of the four statutory classes of § 101. As such, the claimed "computer-readable medium" is not limited to embodiments that fall within a statutory category of invention (*i.e. "storage"* - see Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility - Annex IV(c) (1300 OG 142 signed 26Oct2005). Therefore, claim 25 is rejected as non-statutory.

Claims 26-32 mirror the deficiencies of claim 25 and are also rejected as non-statutory.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-5,7-17,19-28, and 30-33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Scalzi et al. (US Patent 6,075,937) in view of Sachs et al (US Patent 5,560,028).

As to claim 1, Scalzi teaches a method of translating binary code instructions from a source format to a target format for processing by a target processor, said method comprising the steps of:

identifying a source instruction (see column 6, lines 9-10),

selecting a translation template corresponding to said identified source instruction, said template providing a set of target instructions semantically equivalent to said identified source instruction (see column 6, lines 10-11 and column 12, lines 18-21),

translating said identified instruction in accordance with said template (see column 6, lines 11-14), and

outputting said translated instruction for processing by said target processor (see column 6, lines 14-17). Scalzi does not specifically teach generating dependency analysis code including maintaining a counter associated with each instruction to indicate a group number to which the instruction belongs, wherein all instructions of a same group are issued in parallel and assigning each instruction to an earliest group of instructions such that all producers of input and output resources of the instruction have already been assigned to previous groups. In an analogous art, however, Sachs is cited to teach maintaining a counter associated with each instruction (see FIG.14 and associated text, e.g. col.11 lines 3-6) to indicate a group number to which the instruction belongs (see col. 3

lines 22-25), wherein all instructions of a same group are issued in parallel (see col. 3 lines 25-27) and assigning each instruction to an earliest group of instructions such that all producers of input and output resources of the instruction have already been assigned to previous groups (see col.5 lines 10-13 and lines 21-23 – *the input and output resources are dependent on the instruction so in order for the instructions to be executed in parallel, the dependencies have to resolved first; dependent instructions could not be assigned to the same group)*. It would have been obvious to one having ordinary skill in the art at the time of the invention to combine the teachings of Scalzi and Sachs to improve the parallel performance of a program, as disclosed by Sachs (see col.2 lines 55-60).

As to claim 2, Scalzi teaches a method according to claim 1 in which said source and target instructions include a control part and a data part and said control part being used in said identification step to identify an instruction (see column 22, lines 50-52).

As to claim 3, Scalzi teaches a method according to claim 2 in further comprising a transformation step in which said data part from said source instruction is transformed into said corresponding data part or parts of said set of target format instructions (see column 3, lines 2-7).

As to claim 4, Scalzi teaches a method according to claim 3 in which said transformation step is carried out in accordance with a bit filling routine associated with said template (see column 18, lines 14-18 and column 19, lines 46-49 and 57-60).

As to claim 5, Scalzi teaches a method according to claim 4 in which said bit filling routine is uniquely associated with said template (see column 18, lines 14-18 and column 19, lines 46-49 and 57-60).

As to claim 7, Scalzi teaches a method according to claim 2 in which said source instruction control parts are each concatenated to provide a unique identifier and said templates are indexed in accordance with said identifiers (see column 14, lines 31-40).

As to claim 8, Scalzi teaches a method according to claim 7 in which said templates are indexed by said unique identifiers in a look up table (see column 12, lines 56-59).

As to claim 9, Sachs further teaches in which said translation is carried out at runtime of an emulated application program (see paragraph [0149], lines 1-3).

As to claim 10, Scalzi teaches a method according to claim 1 in which said templates are provided by software procedure calls (see column 10, lines 14-17).

As to claims 11, Scalzi in view of Sachs teaches the limitations of claim 1, but does not specifically teach that the source format is 32 bit and the target format is 64 bit. However, Scalzi discloses that the source format of his invention is S/390 and the target format is PowerPC. It is well known in the art that S/390 has a 32-bit architecture and the PowerPC has a 64-bit architecture. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made that the source format is 32-bit and the target format is 64-bit in Scazi's invention, as his method can operate between any platform or processor type.

As to claim 12, Scalzi in view of Sachs teaches the limitations of claim 1, but does not specifically teach that the source format is PA-RISC and the target format is Itanium™ code. Instead, he teaches the source format to be S/390 code and the target format to be PowerPC code. However, it is well known in the art that PA-RISC is a 32-bit architecture and Itanium is a 64-bit architecture, which share the same characteristics as the source and target formats disclosed by Scalzi. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to replace the source and target formats of Scalzi's invention with any other code formats, as Scalzi's method can operate between any platform or processor type.

As to claim 13, Scalzi teaches an apparatus for translating binary code instructions from a source format to a target format for processing by a target processor, the apparatus comprising:

an instruction identifier for identifying a source instruction (see column 6, lines 9-10),

a template selector for selecting a translation template corresponding to said identified source instruction, said translation template comprising a set of target instructions semantically equivalent to said identified source instruction and further comprising input and output resources (see column 6, lines 10-11 and column 12, lines 18-21),

a translator for translating said identified instruction in accordance with said template (see column 6, lines 11-14), and

an output buffer for outputting said translated instruction for processing by said target processor (see column 6, lines 14-17). Scalzi does not specifically teach a scheduler that performs dependency analysis using a counter associated with each instruction to indicate a group number to which the instruction belongs, wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups. In an analogous art, however, Sachs is cited to teach a scheduler that performs dependency analysis using a counter associated with each instruction (see FIG.14 and associated text, e.g. col.11 lines 3-6) to indicate a group number to which the instruction belongs (see col. 3 lines 22-25), wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups (see col.5 lines 10-13 and lines 21-23 – *the input and output resources are dependent on the instruction so in order for the instructions to be executed in parallel, the dependencies have to resolved first; dependent instructions could not be assigned to the same group).* It would have been obvious to one having ordinary skill in the art at the time of the invention to combine the teachings of Scalzi and Sachs to improve the parallel performance of a program, as disclosed by Sachs (see col.2 lines 55-60).

As to claim 14, Scalzi teaches an apparatus according to claim 13 in which said source and target instructions include a control part and a data part

and said instruction identifier uses said control part to identify instruction (see column 22, lines 50-52).

As to claim 15, Scalzi teaches an apparatus according to claim 14 in which in said translator is operable to transform said data part from said source instruction into said corresponding data part or parts of said set of target instructions (see column 3, lines 2-7).

As to claim 16, Scalzi teaches an apparatus according to claim 15 in which said transformation is carried out in accordance with a bit filling routine associated with said template (see column 18, lines 14-18 and column 19, lines 46-49 and 57-60).

As to claim 17, Scalzi teaches an apparatus according to claim 16 in which said bit filling routine is uniquely associated with said template (see column 18, lines 14-18 and column 19, lines 46-49 and 57-60).

As to claim 19, Scalzi teaches an apparatus according to claim 14 in which said source instruction control parts are concatenated to provide a unique identifier and said templates are indexed in accordance with said identifiers (see column 14, lines 31-40).

As to claim 20, Scalzi teaches an apparatus according to claim 19 in which said templates are indexed by said unique identifiers in a look up table (see column 12, lines 56-59).

As to claim 21, Sachs further teaches in which said translation is carried out at runtime of an emulated application program (see paragraph [0047]).

As to claim 22, Scalzi teaches an apparatus according to claim 13 in which said templates are provided by software procedure calls (see column 10, lines 14-17).

As to claims 23, Scalzi in view of Sachs teaches the limitations of claim 13, but does not specifically teach that the source format is 32 bit and the target format is 64 bit. However, Scalzi discloses that the source format of his invention is S/390 and the target format is PowerPC. It is well known in the art that S/390 has a 32-bit architecture and the PowerPC has a 64-bit architecture. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made that the source format is 32-bit and the target format is 64-bit in Scalzi's invention, as his method can operate between any platform or processor type.

As to claim 24, Scalzi in view of Sachs teaches the limitations of claim 13, but does not specifically teach that the source format is PA-RISC and the target format is Itanium™ code. Instead, he teaches the source format to be S/390 code and the target format to be PowerPC code. However, it is well known in the art that PA-RISC is a 32-bit architecture and Itanium is a 64-bit architecture, which share the same characteristics as the source and target formats disclosed by Scalzi. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to replace the source and target formats of Scalzi's invention with any other code formats, as Scalzi's method can operate between any platform or processor type.

As to claim 25, Scalzi teaches a computer program product for translating binary code instructions from a source format to a target format for processing by a target processor, comprising a computer readable medium, further comprising:

a template for use in a binary code translator for translating binary code instructions from a source format to a target format for processing by a target processor (see column 6, lines 10-11 and column 12, lines 18-21), the template comprising:

a template identifier for uniquely associating said template to a source instruction (see column 12, lines 56-59), and

a set of target instructions in a target format semantically equivalent to the source instruction (see column 2, lines 15-20). Scalzi does not specifically teach a set of codes for performing dependency analysis using a counter associated with each instruction to indicate a group number to which the instruction belongs, wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups. In an analogous art, however, Sachs is cited to teach a set of codes for performing dependency analysis using a counter associated with each instruction (see FIG.14 and associated text, e.g. col.11 lines 3-6) to indicate a group number to which the instruction belongs (see col. 3 lines 22-25), wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups (see col.5 lines 10-13 and lines 21-23 – *the input and output resources are dependent on the*

instruction so in order for the instructions to be executed in parallel, the dependencies have to resolved first; dependent instructions could not be assigned to the same group). It would have been obvious to one having ordinary skill in the art at the time of the invention to combine the teachings of Scalzi and Sachs to improve the parallel performance of a program, as disclosed by Sachs (see col.2 lines 55-60).

As to claim 26, Scalzi teaches a computer program product according to claim 25 further comprising a set of codes causing a computer to derive template identifier from a control part of the source instruction (see column 22, lines 50-52, and lines 57-61).

As to claim 27, Scalzi teaches a computer product according to claim 26 wherein the template causes a computer to transform a data part of the source instruction into at least one corresponding data part of the set of target instructions (see column 3, lines 2-7).

As to claim 28, Scalzi teaches a computer product according to claim 27 further comprising a set of codes for causing a computer to bit fill the data part of the source instruction.

As to claim 30, Scalzi teaches a computer product according to claim 26 wherein the template causes a computer to create the template identifier by concatenating the control part of said source instruction (see column 14, lines 31-40).

As to claim 31, Scalzi in view of Sachs teaches the limitations of claim 25, but does not specifically teach that the template causes a computer to transform

a source instruction having a 32 bit format to a target instruction having a 64 bit format. However, Scalzi discloses that the source format of his invention is S/390 and the target format is PowerPC. It is well known in the art that S/390 has a 32-bit architecture and the PowerPC has a 64-bit architecture. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made that the source format is 32-bit and the target format is 64-bit in Scalzi's invention, as his method can operate between any platform or processor type.

As to claim 32, Scalzi in view of Sachs teaches the limitations of claim 25 but does not specifically teach that the template causes a computer to transform PA-RISC source code into Itanium TM target code. Instead, Scalzi teaches the source format to be S/390 code and the target format to be PowerPC code. However, it is well known in the art that PA-RISC is a 32-bit architecture and Itanium is a 64-bit architecture, which share the same characteristics as the source and target formats disclosed by Scalzi. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to replace the source and target formats of Scalzi's invention with any other code formats, as Scalzi's method can operate between any platform or processor type.

As to claim 33, Scalzi teaches a computer program product for translating binary code instructions from a source format to a target format for processing by a target processor comprising a computer readable medium comprising:

a first set of codes for causing a computer to identify a source instruction (see column 6, lines 9-10),

a second set of codes for causing a computer to select a translation template corresponding to said identified source instruction said template providing a set of target format instructions semantically equivalent to said identified source instruction (see column 6, lines 10-11 and column 12, lines 18-21),

a third set of codes for causing a computer to translate said identified instruction in accordance with said template (see column 6, lines 11-14), and

a sixth set of codes for causing a computer to output said translated instructions (see column 6, lines 14-17). Scalzi does not specifically teach a fourth set of codes for performing dependency analysis using a counter associated with each instruction to indicate a group number to which the instruction belongs, wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups. In an analogous art, however, Sachs is cited to teach a fourth set of codes for performing dependency analysis using a counter associated with each instruction (see FIG.14 and associated text, e.g. col.11 lines 3-6) to indicate a group number to which the instruction belongs (see col. 3 lines 22-25), wherein the counter computes the earliest group to which a given instruction can be assigned such that all procedures of inputs and outputs of the instruction have already been assigned to previous groups (see col.5 lines 10-13 and lines 21-23 – *the input and output resources are dependent on the instruction so in order for the instructions to be executed in parallel, the dependencies have to resolved first;*

dependent instructions could not be assigned to the same group). It would have been obvious to one having ordinary skill in the art at the time of the invention to combine the teachings of Scalzi and Sachs to improve the parallel performance of a program, as disclosed by Sachs (see col.2 lines 55-60).

6. Claims 6 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Scalzi et al. (US Patent 6,075,937) in view of Sachs et al (US Patent Application 5,560,028) as applied to claims 1 and 13 above, and further in view of Lee (US Patent 5,828,884).

As to claim 6, Scalzi in view of Sachs teaches the limitations of claim 3, but does not specifically teach the transformation of data of one type of endianness to data of another type of endianness. Lee is cited to teach a method for converting data between different endian formats similar to Scalzi's. Lee teaches a method for compiling a software program and executing the program on a system that converts data between little endian and big endian formats (see Abstract). It would have been obvious to one having ordinary skill in the art at the time the invention was made to combine the teachings of Scalzi in view of Sachs with those of Lee because Lee provides a method that allows software developers to develop more efficient, portable, and bug-free code with respect to byte ordering issues.

As to claim 18, Scalzi in view of Sachs teaches the limitations of claim 15, but does not specifically teach the transformation of data of one type of endianness to data of another type of endianness. Lee is cited to teach a method

for converting data between different endian formats similar to Scalzi's. Lee teaches a method for compiling a software program and executing the program on a system that converts data between little endian and big endian formats (see Abstract). It would have been obvious to one having ordinary skill in the art at the time the invention was made to combine the teachings of Scalzi in view of Sachs with those of Lee because Lee provides a method that allows software developers to develop more efficient, portable, and bug-free code with respect to byte ordering issues.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Cheneca P. Smith whose telephone number is (571) 270-1651. The examiner can normally be reached on Monday-Friday 7:00-4:30 EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

CS
2/6/2008

/Tuan Q. Dam/

Supervisory Patent Examiner, Art Unit 2192